ESSENTIAL
AN ODS NETWORKS COMPANY

SGI PCI HIPPI Interface Controller Card Drivers and Utilities for Octane and Origin 200/2000

Release 1.0.4 for IRIX 6.5

Thank you for purchasing Essential Communications Corp.'s HIPPI Network Interface Card (NIC). Instructions are included in this sheet for downloading the appropriate driver and where to get support in case you have any problems or questions. Please ensure that you have loaded the latest patches for your operating system in conjunction with loading this driver. Please contact Essential if you have any problems with this driver.

INSTRUCTIONS FOR USE:

You need an FTP utility to download the driver, utilities, and all further documentation. Our FTP Site is located at ftp.esscom.com. Instructions for the driver and utilities are also included in the manual.

Please download the following files in binary mode:

/release/fkpds/H_IP27IP30_6.5_1.0.4.tar
/release/fkpds/readme.txt

You must cd into the parent directory before doing a get on the filenames.

Please follow the instructions in the readme.txt file before installing the software. This file contains all the information necessary to install and configure the software, as well as the release notes for this driver.

Please contact support at help@ods.com or 800-832-6637 if you have any questions or problems. You may also find information on our web site at http://www.ods.com at the "Customer Support" option.

Essential/ODS Networks 4239 Balloon Park Road, NE Albuquerque, NM 87109
Telephone: (505) 344-0080 Fax: (505) 344-0408 www.ods.com

Essential/ODS Networks

Limited One Year Warranty and Support Policies

Limited Hardware Warranty: Essential Communications (Essential) warrants that each Essential hardware product will be free from material defects for a period of one (1) year from the date of shipment. For purpose of this warranty, "material defect" shall mean any defect in an Essential product that can be reproduced by Essential and is a substantial nonconformity from the published specifications for such Essential hardware products. Your sole remedy and Essential's sole responsibility under this warranty shall be repair or replacement of the Essential hardware product within 5 working days after receipt of defective material. Replacement products may be refurbished or contain refurbished material but will be warranted.

The foregoing warranty applies only to the original purchaser of the Essential hardware product. Essential shall have no obligation to provide warranty services if the defect in the Essential hardware product is caused by accident, operator error, malfunction of non-Essential hardware or software, modifications of the Essential hardware product, or use of the Essential hardware product that is not in accordance with the operating instructions for the Essential hardware product.

Limited Software Warranty: Essential warrants that if the software fails to substantially conform to the specifications and environment in the documentation and if the nonconformity is reported to Essential in writing within 90 days from the date the license is purchased, then Essential will make best efforts to remedy the nonconformity or offer to refund the license fee upon return of all copies of software and documentation to Essential. In the event of a refund, your right to use the software shall automatically expire.

Essential warrants that if the magnetic media or documentation are in a damaged or physically defective condition at the time the license is purchased and if they are returned to Essential (postage prepaid) within 30 days of purchase, Essential will provide replacements at no charge.

Essential shall provide limited post installation phone support for a period of 30 days from shipment.

EXCEPT FOR THE EXPRESS WARRANTY STATED HEREIN, ESSENTIAL DISCLAIMS ALL OTHER EXPRESS OR IMPLIED WARRANTIES REGARDING THE ESSENTIAL PRODUCTS, INCLUDING WITHOUT LIMITATION ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Warranty Service

NOTE: Prior to returning any hardware product manufactured by Essential for warranty or support service, complete the following steps:

1. Review all User's Manuals and product READ-ME files. Record all messages and conditions that might help identify the specific hardware or software problem.

2. Contact Essential's Technical Support Center at **(505) 344-0080** or e-mail **support@esscom.com** and provide us with the hardware product model and serial number and a description of the problem you are experiencing.

3. Once a problem has been determined by Essential's Technical Support, service options for repair will be governed by the warranty policies. A Return Material Authorization (RMA) number will be required as described below.

Upon discovery of a material defect, call Essential's Technical Support at (505) 344-0080 to obtain a Return Material Authorization (RMA) number. Please return the Essential hardware product, postage prepaid, labeled with the RMA number, to:

Essential Communications
Attn: RMA Desk
1551 Mercantile NE, Suite A
Albuquerque, NM 87107

Essential's sole responsibility under this warranty shall be, at Essential's option, to either repair or replace the Essential product. All defective Essential hardware products, or defective components thereof, returned under this warranty shall become Essential's property. If Essential determines that the original Essential hardware product did not contain a material defect, you, the customer, shall pay Essential all costs of handling, transportation, and diagnostic time at Essential's prevailing rates.

Liability Limitations

ESSENTIAL'S LIABILITY ARISING OUT OF THE SALE OF THE ESSENTIAL HARDWARE PRODUCTS OR THE REVISIONS OF WARRANTY OR SUPPORT SERVICES SHALL BE LIMITED TO THE AMOUNT PAID FOR THE ESSENTIAL PRODUCTS OR THE PROVISIONS OF WARRANTY OR SUPPORT SERVICES. IN NO EVENT SHALL ESSENTIAL BE LIABLE FOR ANY SPECIAL, CONSEQUENTIAL, INCIDENTAL, OR INDIRECT DAMAGES, HOWEVER CAUSED, ON ANY THEORY OF LIABILITY, AND WHETHER OR NOT ESSENTIAL HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. THESE LIMITATIONS SHALL APPLY NOTWITHSTANDING ANY FAILURE OF ESSENTIAL PURPOSE OF ANY LIMITED REMEDY.

Contact Essential/ODS Networks:

Telephone: (505) 344-0080; Fax: (505) 344-0408
E-mail: support@esscom.com (for technical support); info@esscom.com (for product information); sales@esscom.com (for product pricing & quotes)
Worldwide WEB URL: http://www.esscom.com/

Essential/ODS Networks
Products Software License

The software received with this product is protected by the copyright laws of the United States and international copyright treaties. By accepting this Essential product you have agreed to the terms and conditions as stated below. Essential retains all rights not expressly granted to you.

Protection - You agree to take reasonable steps to protect the software and documentation from unauthorized copying or use. The software source code represents and embodies trade secrets of Essential and/or its licensers. The source code is not licensed to you. You agree not to disassemble or de-compile the software in order to discover the source code.

Copies and Adaptations - You may only adapt or copy as an essential step in the utilization of the software, or to make an archival or back-up copy of the software. All proprietary rights notices must be faithfully reproduced and included on all copies and adaptations. Unless explicitly stated in the documentation, you may not copy the documentation.

Ownership - Ownership of, and title to, the software and documentation (including any adaptations or copies) is and shall remain with Essential and/or its licensers. Copies are provided to you only to allow you to exercise your rights under this license. Only this license and the right to use herein is purchased by you.

Restrictions - Except as expressly authorized in this agreement you agree not to rent, lease, sublicense, distribute, transfer, copy, reproduce, display, modify, or time share the software or documentation.

Transfer of License - You may transfer this license to another individual or entity if you obtain prior written consent from Essential. If Essential approves the transfer of the license, you agree to transfer all copies of the software and documentation, including any copies or adaptations you have made.

Governing Law - This license shall be governed by, and interpreted in accordance with, the laws of the state of New Mexico and the United States of America (excluding their laws governing conflicts of law).

Complete Agreement - This license is the entire understanding between you and Essential and may be amended only in writing signed by both parties.

Waiver - No waiver of any right under this license shall be effective unless in writing, signed by a duly authorized official of Essential. No waiver of any past or present right arising from any breach or failure to perform shall be deemed to be a waiver of any future right arising under this license.

Severability - If any provision of this license is invalid or unenforceable, that provision shall be construed, limited, modified or, if necessary severed, to eliminate its invalidity or unenforceability, and the other provisions shall be interpreted to give effect to the intent of the severed provisions.

Export - The United States Government strictly controls the exportation of technical materials. You agree that you will not knowingly, directly or indirectly, without prior written consent, if required, of the Office of Export Licensing of the U.S. Department of Commerce, Washington D.C. 20230, export or transmit the software to any country to which such transmission is restricted by applicable regulations or statutes.

United States Government Restricted Rights - Use, duplication, or disclosure by the United States Government is subject to restrictions as set forth in FAR 52.227-14 (June 1987) Alternate III(g)(3) (June 1987), FARS 52.227-19 (June 1987), DFARS 52.227-7013 (c)(1)(ii) (June 1988), as applicable. Contractor/Manufacturer is Essential Communications, 4374 Alexander Blvd. NE, Suite T, Albuquerque, New Mexico 87107.

ESSENTIAL/ODS Networks
OWNER REGISTRATION FAX

1551 Mercantile NE, Suite A
Albuquerque, NM 87107
Fax: 505-344-0408

Name_____ Title_____ Email:_____

Company_____ Street Address_____

City_____ State_____ Zip_____ Tel_(_____)_____ Fax_(_____)_____

Product Purchased: _____ Serial Number: _____

Date of purchase : _____

What is the serial # of your Essential product?_____

What equipment is your Essential product connected to? _____

For what primary applications?_____
Whom shall we contact for upgrade and product revision Information? Name:_____ Tel. _(_____)_____
Email: _____

PLEASE COMPLETE AND FAX THIS REGISTRATION TO: 505-344-0408

HIPPI
Network Interface Card

SGI PCI and GIO

Installation Guide

Version 1.5

ALBUQUERQUE, NEW MEXICO

© 1997.
All rights reserved under Pan-American and International Copyright Conventions.

AlphaStation, AlphaServer and the digital logo are trademarks of Digital Equipment Corporation.

Essential Communications
4374 Alexander Blvd. NE, Suite T
Albuquerque, NM 87107
Tel: +1-505-344-0080
Fax: +1-505-344-0408
E-mail: info@esscom.com
Website: http://www.esscom.com

Part Number: 9931-1125-0000

FEDERAL COMMUNICATIONS COMMISSION (FCC) STATEMENT

This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to Part 15 of the FCC rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio-frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference, in which case the user will be required to correct the interference at his or her own expense.

CAUTION Changes or modifications to this equipment not expressly approved by the manufacturer responsible for compliance could void the user's authority to operate this equipment.

TABLE OF CONTENTS

Chapter 1 Introduction 5
- 1.1 Hardware Overview 5
- 1.2 Software Overview 5
- 1.3 Introduction to HIPPI Networking 6
- 1.4 Standards Documents 6
- 1.5 Switched Network Topology 7
- 1.6 I-Field 7
- 1.7 Framing Protocol 9

Chapter 2 Hardware Installation 11
- 2.1 Before You Begin 11
- 2.2 Hardware Installation Procedure 12
  - 2.2.1 Preparing the Chassis 13
  - 2.2.2 Opening the SGI Workstation Chassis 13
  - 2.2.3 Installing the HIPPI Card 13
  - 2.2.4 Closing the Chassis 14

Chapter 3 Software Installation 15
- 3.1 Installation Prerequisites 15
- 3.2 Retrieving the Software 16
- 3.3 Installing the Software 17
- 3.4 Common Installation Problems 18
- 3.5 Removing the Software 18
- 3.6 On-line Support 19

Chapter 4 Software Configuration 21
4.1 Initialization Files 21
4.2 Firmware Installation 23

Chapter 5 Essential HIPPI Driver Utilities 25
5.1 essarp 25
5.2 essd 27
5.3 esshippi 28
5.4 blast 30
5.5 sink 32
5.6 tsock 33
5.7 essdmpd 40

Chapter 6 Administrative Utilities 43
6.1 essdb 43
6.2 essstat 44
6.3 esstb 45
6.4 esstune 45
6.5 essgidle 47
6.6 essdnld 47
6.7 essdisp 48

1

INTRODUCTION

The information in this document is provided to assist you with installation of a HIPPI network interface card (NIC) in a Silicon Graphics system, and connect the system to a HIPPI network.

1.1 Hardware Overview

The Essential HIPPI NIC provides data transfer at rates of up to 800 megabits per second over incoming and outgoing links simultaneously. The Essential GIO HIPPI adapter requires a single, full-length GIO slot in the host workstation. The PCI adapter requires a single empty PCI slot.

1.2 Software Overview

The Essential HIPPI driver is loaded onto the host system as part of the NIC installation; the Indigo^2 must be running IRIX version 6.2 or later. The O2 system must be running 6.3 or later. The Octane or Origin200 must be running IRIX version 6.4. The driver provides a network interface for TCP/IP, providing the IF layer of a BSD 4.3 sockets protocol stack. You can use `ifconfig` to control the HIPPI network interface, and you can use standard network utilities, such as `netstat`, to gather network statistics. Standard TCP/IP network programs, such as Telnet and FTP, can be operated over the HIPPI network.

The Essential TCP/IP implementation is fully compliant with RFC 1374: "IP and ARP on HIPPI."

In addition, a character device interface lets you access the HIPPI device through standard device-driver calls, such as `open()`, `read()`, `write()`, `select()`, and `ioctl()`.

1.3 Introduction to HIPPI Networking

HIPPI (High Performance Parallel Interface) was developed in the late 1980s to provide high-speed I/O connectivity between supercomputers and other devices. HIPPI has evolved beyond this original purpose to encompass high-speed local-area networking.

The basis of HIPPI is a simplex, 100-megabyte per second (MBps), or 800 megabit per second, channel. As on Essential's NICS, two HIPPI channels are usually combined to create a duplex connection.

Initially, the only link medium for HIPPI was parallel copper cables that could be up to 25 meters long. It is now possible to establish multimode and single-mode fiber connections that can be up to 10 kilometers long.

1.4 Standards Documents

HIPPI consists of several protocol layers, each of which is defined by a standards document.

ANSI X3.183-1991 defines the HIPPI physical layer, referred to as HIPPI-PH. This document defines the basic copper or fiber-optic cables, the interconnect mechanisms, and the low-level movement of data. (Data is broken up into packets and further subdivided into bursts of up to 256 four-byte words.)

There are three layers on top of the physical medium to allow the basic HIPPI channel to be used in a very efficient, low-latency network. These are HIPPI-SC (switch control, **ANSI X3.222-1993**), HIPPI-FP (framing protocol, **ANSI X3.210-1992**), and HIPPI-LE (link encapsulation, **ANSI X3.218-1993**). The ANSI Serial-HIPPI Specification defines fiber-optic implementations.

These ANSI standards documents, along with **RFC 1374**, define how a TCP/IP network can operate on top of a physical network of HIPPI channels and switches. This mode of operation allows other protocols and proprietary applications to access the HIPPI network.

1.5 Switched Network Topology

The HIPPI network itself consists of a number of systems that have duplex HIPPI connections and are interconnected with a set of high-speed switches. The HIPPI connections in these systems are either provided by add-on boards (for example, the Essential GIO-HIPPI board) or are intrinsic parts of the system (for example, the Cray C90 HIPPI). Host software allows you to set up the HIPPI channel in a network orientation.

The switches used in HIPPI networks are non-blocking with very low latency. The total throughput of a HIPPI switch is the total of the HIPPI connections on the switch multiplied by two. For example, a 16 × 16 switch has 32 connections, for a total throughput of 32 × 100 MB/s, or 3200 MB/s.

To move data through the switch network to a specific destination, a special header called an "I-field" is used. I-field format is defined in the HIPPI-PH document, and I-field-based switching is defined in the HIPPI-SC document.

1.6 I-Field

The **I-field** defines destination routing for a data packet. The I-field is four bytes long and carries two sequences of information: a control byte and a network/switch route. The control byte tells the switch how to interpret the network/switch route portion of the I-field.

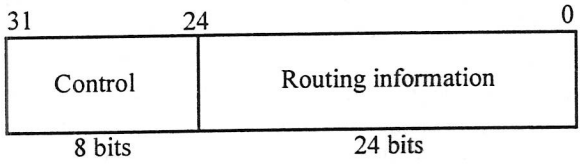

Figure 1.1. The I-field

Table 1 defines the information fields within the routing control bytes.

Table 1.

Bit(s)	Designation	Description
29-31	Reserved	

Table 1.

Bit(s)	Designation	Description
28	Double-wide mode	Distinguishes between 32- or 64-bit connections when using a 64-bit port.
27	Direction	Identifies which end of the I-field routing information to use (0 for right, 1 for left).
25-26	Path selection	The two bits for path selection are defined as follows: 1,1 Logical address with automation and random selection of one port from a list. 1,0 Reserved for future use. 0,1 Logical address with automatic selection of the first available port to reach destination. 0,0 Source routing.
24	CAMP_ON	If CAMP_ON is set to 1, the connection waits for the destination to be available. If set to 0, the connection is rejected if the requested destination is not available.

There are two basic forms of HIPPI switch addresses: source route addresses and logical addresses. The **source route address** uses the three routing bytes to guide the packet through the switch network. The least-significant three nibbles (12 bits) indicate which of the switch's ports the data should be switched onto. The switch then rotates the routing field three nibbles to the right so that the next switch (if there is one) sees the next three nibbles as the port it should use. The switch also places the incoming port in the most-significant three nibbles. This allows the final end point to create a "reverse route" from the I-field that it receives. See the HIPPI-SC document for additional information.

The **logical address** is a more elegant method of routing data. This mechanism takes advantage of the switch's internal routing tables. You can think of the logical address in the same way as you think of an IP address, and the switch's routing table as the routing table found in an IP router. When the switch sees a packet with the logical address flag set, it looks in its routing table and determines which port should send the packet out. This is repeated in all the switches until the packet arrives at its final and proper destination.

Another important I-field control bit is the **camp-on** field. This field allows the switch to keep a packet waiting while other data is passing through a final or intermediate destination that the waiting packet wishes to use. This allows a guaranteed arrival when the end destination is active.

1.7 Framing Protocol

Because the HIPPI network can carry different types of traffic, a framing protocol was developed. This protocol allows the destination to determine what type of traffic it is receiving. The most common framing protocol used is ISO 802.2. This protocol is typically used to carry TCP/IP traffic. Other protocols used are IPI3 (for disk device traffic) and proprietary protocols.

The value in the ULP (upper-layer protocol) field of the framing protocol determines the protocol being carried. Values include those for a number of well-known protocols (for example, 802.2 is 4 and IPI3 are 6, 7, and 8) and a "local-use" bit that allows you to specify that the protocol is a local one.

2

HARDWARE INSTALLATION

This chapter describes how to unpack, handle and install the Essential HIPPI network interface card into an SGI workstation. There are currently two varieties of card which can be installed in an SGI workstation. The GIO card for the Indigo^2 workstation and the PCI card for the Octane, Origin200, or O2 workstation. Please read all information in this chapter carefully *before* installing the HIPPI card.

2.1 Before You Begin

During installation, you will need the following tools:

- A #2 Phillips screwdriver to install the lock-down screw.
- A grounding strap. Like any other electronic equipment, the HIPPI NIC is sensitive to electro-static discharge (ESD). Take care to avoid damaging the card by mishandling it.

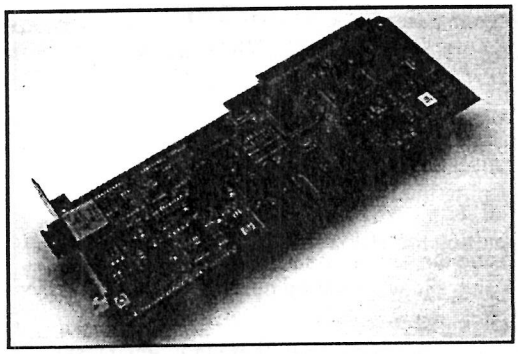

Figure 2.1. The GIO-HIPPI card.

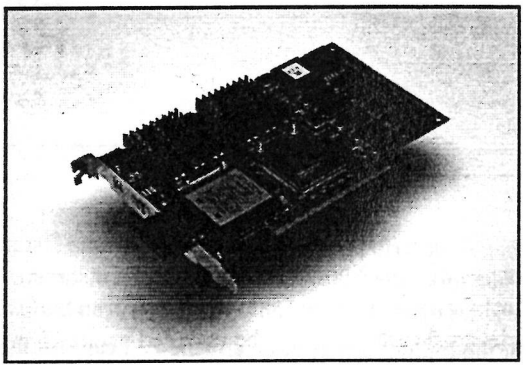

Figure 2.2. The PCI HIPPI Card

Before beginning installation, without removing the Essential HIPPI NIC from its antistatic bag, inspect the product for any signs of obvious damage, such as chipped or loose components. Contact Essential if the NIC is damaged.

Finally, read through the remainder of the installation procedure before you begin, to familiarize yourself with the process. This will simplify the installation process.

2.2 Hardware Installation Procedure

Installing the Essential HIPPI NIC in an SGI system involves the following tasks:

- Preparing the workstation chassis.
- Opening the workstation chassis.
- Installing the HIPPI adapter.
- Closing the workstation chassis.

The following sections detail these individual tasks.

2.2.1 PREPARING THE CHASSIS

To prepare the SGI chassis before installing the HIPPI NIC, perform the following steps:

1. Shut down the system and turn off the power. Refer to your workstation documentation for information about shutting down the system.

To avoid personal injury and equipment malfunction, never attempt to add or remove any card from a workstation without first turning off power to the machine.

2. Isolate the chassis from the AC power source by unplugging the power cord.
3. Wait 15 seconds, to allow the power-supply capacitors to discharge fully.

2.2.2 OPENING THE SGI WORKSTATION CHASSIS

Refer to your workstations documentation before opening the chassis. Please follow all instructions described in that manual including any precautions mentioned in the text.

2.2.3 INSTALLING THE HIPPI CARD

Refer to your workstation documentation for more information about installing network cards. To install the Essential HIPPI card in the workstation, follow these steps:

1. Remove the HIPPI adapter from its antistatic bag.

Handle circuit cards only while wearing a wrist strap. This prevents potential damage to the card from static electricity.

2. Select an open slot in which to install the adapter. The slot you choose will depend on which other cards if any you already have installed.
3. in the O2 workstation there is only one PCI slot. You will need to remove the PCI tray from the system module before following the next step (see the Users Guide for the workstation for instructions).
4. Remove the lock-down screw holding the metal port cover in place.
5. Remove the metal port cover.

6. Gently slide the HIPPI card into the card guides and the selected slot.
7. Press the end of the NIC to firmly seat it in the GIO or PCI slot. Be sure the card is completely inserted into the slot.
8. Lock down the NIC bracket using the lock-down screw you removed from the port cover.
9. In the O2 you will need to reinstall the PCI tray into the system module before putting the system module back into the chassis.

2.2.4 CLOSING THE WORKSTATION CHASSIS

To close the workstation or server, please follow the steps indicated in the worksations documentation. Remeber to not reconnect power or any other cords until chassis has been closed and secured..

You are now ready to install the Essential HIPPI driver software. This operation is described in Chapter 3.

3

SOFTWARE INSTALLATION

This chapter describes installing the Essential HIPPI driver software onto an SGI Indigo^2, Octane, Origin200, or O2 workstation. Installation prerequisites are listed, and the steps required to install the driver and rebuild the kernel are described. Finally, verifying that the adapter and driver have been installed correctlyis explained.

3.1 Installation Prerequisites

Before installing the Essential HIPPI driver, ensure you have the following hardware and software:

- An SGI Indigo^2 system, with an Essential GIO HIPPI NIC installed, running IRIX version 6.2 or later, an SGI Octane, with an Essential PCI HIPPI NIC, running IRIX version 6.4, an SGI Origin200, with an Essential PCI HIPPI NIC, running IRIX version 6.4, or an SGI O2 system, with an Essential PCI HIPPI NIC installed, running IRIX version 6.3 or later.
- The correct version of the Essential HIPPI driver (to find out which version to use read the README.sgi file as described below).
- Essential NIC drivers are downloaded via FTP from Essential's Internet site. Therefore, you must have a system with Internet network access.
- You must have `root` (superuser) access on the SGI system.

Before you install the driver software, create a working directory for use during the software installation; for example: **`/usr/ess`**. In the following procedural description, the working directory is referred to as "`working_directory`," thus:

```
mkdir working_directory
```

3.2 Retrieving the Software

The Essential HIPPI driver software is a tar file that is available via Anonymous FTP.

USING FTP

To retrieve the software using FTP, follow this procedure:

1. Change to the working directory you just created:

   ```
   cd working_directory
   ```

2. Connect to Essential's FTP site:

   ```
   ftp ftp.esscom.com
   ```

3. Log in with the username `anonymous`. Enter your full e-mail address as the password.

4. Copy the files from the proper installation directory:

   ```
   Please note:  The directory structure constantly
   changes.  Please refer to the customer letter for
   the directory containing your installation files.
   ```

5. Before downloading the tar file, switch to binary transfer mode:

   ```
   ftp> bin
   ```

6. Retrieve the file (named in the customer letter) containing the Essential GIO HIPPI or PCI HIPPI driver software:

   ```
   ftp> get filename
   ```

7. Quit FTP:

   ```
   ftp> quit
   ```

3.3 Installing the Software

After you have installed the Essential HIPPI NIC into your SGI system, and retrieved the driver software, you must install the driver software. To do this, follow these steps (you must have system `root` priviledges):

1. Be sure the current directory is your working directory:

   ```
   cd working_directory
   ```

2. If you retrieved the installation files from the FTP site, you will need to "untar" them.

   ```
   tar -xvf filename
   ```

3. Launch the Software Manager.

   ```
   swmgr
   ```

4. Enter <working_directory> in the *Available Software* text box.

   ```
   <working_directory>
   ```

5. Click the *Lookup* button or press the Return key on your keyboard. The Software Manager will look for a distribution package, and the *Start* button will be highlighted.

6. Select "Customize Installation." The Essential HIPPI installation package should appear in the main window.

7. Select the Essential HIPPI installation package under the *Install* heading; a check will appear in the checkbox.

8. Begin the installation by clicking the *Start* button.

9. When the Software Manager has completed the driver installation, choose the **Quit** command from the File menu to close the Software Manager.

10. Reboot the system:

    ```
    reboot
    ```

    You have now completed installation of the Essential HIPPI driver software.

3.4 Common Installation Problems

Unknown Message: 3 a 0 <random hexadecimal value>

Probable Cause: The cables are probably swapped or you have specified a bad I-field.

Solution: Verify the I-field and check that the cables are not reversed before continuing.

ioctl failed: Device busy

Probable Cause: The IP layer is currently up and running.

Solution: Disable the IP layer network device by running

% ifconfig ess0 down

% esshippi off

3.5 Removing the Software

To remove the Essential HIPPI driver software from an SGI system, follow these steps:

1. Launch the Software Manager.

   From the desktop control panel, open the System menu and choose **Software Manager**.

2. Select *Manage Installed Software*.

3. Select the correct Essential driver software from the installed software list.

4. Click the *Start* button to begin removal of installed software.

5. Exit the Software Manager.

6. Replace the existing kernel with one that does not include the ess driver.

7. You can either rebuild and install a new kernel, or you can rename the `/unix.orig` kernel to `/unix`.

8. Reboot the system:

   `reboot`

3.6 On-line Support

When you register your software acquisition with us, you are eligible for 30 days of telephone and e-mail support (1-505-344-0080 and support@esscom.com, respectively).

Also, fixes for known problems are posted on our Website; go to http://www.esscom.com/support/. If you are experiencing difficulty, check that page first.

4

SOFTWARE CONFIGURATION

This chapter describes how to configure the HIPPI driver once installed. Please read any release notes or readme files accompanying the driver for the latest information about software configuration before proceeding with the steps listed in this chapter.

4.1 Initialization Files

First make sure that /etc/init.d/Essnic.pre and Essnic.post and other files include the necessary information to fully configure the machine to be able to use TCP/IP for communication:

1. Modify /etc/init.d/Essnic.pre file to include necessary information

You will need an essarp -l statement with the information about the Logical Switch address of the machine you are installing on. This information is found in the switch configuration itself, and should be unique to each HIPPI NIC.

Example:

/etc/essarp -l 00e

Your Essnic.pre may have the following statement:

/etc/essgidle 1

If so then change the value to a number between 2 and 10 for performance improvements. This statement is only valid for the Indigo^2 workstation. The best value will vary based on the particular graphics card and processor you are using. See Chapter 6 for more information on the essgidle utility. View the release notes that came with the driver for more information on the best value for the particular workstation you are using. If you are on any PCI based workstation, you will not use essgidle.

2. Modify /etc/init.d/Essnic.post file to include necessary information

You will need to set the ULA 's and logical addresses of the other machines on the hippi network as well as the machine you are on.

Example:

essarp -s george-h 0:a0:88:0:10:63 00e ess0

essarp -s barney-h 0:a0:88:0:10:66 102 ess0

3. Change /etc/hosts to include IP address and names of hippi machines:

Example:

196.14.231.17 george-h.esscom.com george-h

4. Modify /etc/config/netif.options

add two lines based on how many network devices are already present:

Example:

if2name=ess0

if2addr=george-h

This assumes that you already have an if1name and if1addr for an Ethernet connnection. Again this will depend on how many network cards you have in the machine.

5. Add file /etc/config/ifconfig-x.options

based on how the numbering is done in netif.options create a file for the hippi interface. For example if you used the previous example you would create a file called "ifconfig-2.options" then you would place

the arguments to ifconfig in the file:

Example:

netmask 0xffffff00 up

You should then reboot the machine and the network should be set up.

4.2 Firmware Installation

You now need to install new firmware to support the latest version of the driver. There is an automated script which now turns off the driver firmware; downloads the new EEPROM runcode; and tunes the driver based on our best performance parameters. As root run the following command (after the software installation process is completed and the machine has been rebooted):

/usr/Essential/hippi/scripts/esssetup.sh

There are no differences in the command structure itself for any of the varying SGI drivers. Tuning parmeters are specific to the platform themselves, and differ for each hardware and software revision.

There should not be any platforms that are not accounted for in the tuning. However, if you would like assistance in tuning the driver and firmware for specific applications, or just have questions on what the settings for these parameters are, please contact Essential Support at support@esscom.com or call 505-344-0080 opt. 2. See Chapter 6 for more information about how to use esstune or the other utilities.

After firmware installation reboot the machine. The program essdnld downloads the firmware to the card but does not load it into the cache, rebooting will accomplish this task. See Chapter 5 and 6 for a further explanation of all of these utilities.

5

ESSENTIAL HIPPI DRIVER UTILITIES

The Essential HIPPI driver software includes the following user-level utilities and diagnostics. This chapter provides the reference manual pages for these commands:

- **essarp** controls and displays HIPPI address resolution.
- **essd** provides a process context for the ARP agent to broadcast ARP requests to known hosts.
- **esshippi** is a HIPPI control and status utility.
- **blast** is a HIPPI Character driver transmitter example.
- **sink** is a HIPPI Character driver receiver example.
- **tsock** is a socket-based network exerciser.
- **essdmpd** is a HIPPI dump Daemon.

5.1 essarp

NAME	essarp	Essential HIPPI ARP (Address Resolution Protocol) display and control
SYNOPSIS	essarp hostname	Displays the Internet-to-HIPPI address translation table entry used by the Address Resolution Protocol for HIPPI for the specified hostname.
	essarp -a [unit]	Displays all current ARP entries in the kernel table.
	essarp -h [unit]	Displays the HIPPI adapter's ULA and logical address.
	essarp -s [unit]	hostname adapter-ULA I-field [interface]

essarp -d hostname Delete the ARP entry for that hostname

essarp -D logical address Delete ARP entry of indicated logical address.

essarp -c [unit] Clear the ULA to Logical Address mapping table for the "unit" of non-reserved and non-permanent entries.

essarp -t [unit] Build the ULA to Logical Address mapping table by using self-discovery.

essarp -l logical address [unit] Set adapter switch Logical Address.

essarp -i [unit] Invalidate the adapter Logical Address.

DESCRIPTION

The essarp utility displays and modifies the Internet-to-HIPPI address translation tables used by the Address Resolution Protocol (ARP) for HIPPI (see RFC 1374). This utility is useful in providing ARP address translation information for hosts that do not support ARP over HIPPI.

In particular, essarp allows you to manage the IEEE ULA (universal LAN address), as well as the HIPPI switch address (I-field).

When a destination does not support ULA (the ULA is 0:0:0:0:0:0), essarp creates a locally administered ULA that uses the logical address as the low-order 12 bits of the ULA. IP packets directed to the host are sent with a ULA of 0:0:0:0:0:0 in both the source and destination fields. The Essential HIPPI driver always accepts packets with this ULA in accordance with the HIPPI-LE specification.

Essarp supports multiple cards. In the above commands the "unit" indicates which card (hip0, hip1, etc.).

OPTIONS

-a Displays all current ARP entries in the kernel table.

-h Displays the HIPPI adapter's ULA and logical address.

-s Create an ARP entry for the host hostname that has the universal LAN address (ULA) adapter-ULA, the HIPPI switch address I-field, and, optionally, the network interface. You specify the ULA as three hexadecimal char-

acters (for example, 3EF). The network interface is the name given by the netstat -i command (for example, ess0). If an ARP entry already exists for the specified hostname, the existing entry is updated with the new information. The entry is permanent. Using this command requires superuser privileges.

-d If an ARP entry exists for the host hostname, delete the entry. Using this command requires superuser privileges.

-D If an ARP entry exists for the logical address, delete the entry. Using this command requires superuser privileges.

-c Clear the ULA to Logical Address mapping table for the "unit" of non-reserved and non-permanent entries. This returns EBUSY when the table is being updated. Permanent entries may be deleted using the '-d ' option.Using this command requires superuser privileges.

-t Build the ULA to Logical Address mapping table by sending a HIPPI self discovery response packet to each non-reserved and non-excluded logical address. This returns EBUSY when the table is being updated. Using this command requires superuser privileges.

-l Set the adapter switch Logical Address. If the adapter discovers itself at a different logical address, the discovered address is used. Using this command requires superuser privileges.

-i Invalidate the adapter logical address. Address discovery is performed if the unit is "ifconfig up". Using this command requires superuser privileges.

SEE ALSO arp, essd

5.2 essd

NAME	essd	Essential HIPPI ARP (Address Resolution Protocol) agent daemon.
SYNOPSIS	essd	Starts the essd daemon.

essd -h hostname Set the given host name into the driver.

essd -k Kill the daemon. This is the only way to kill essd.

DESCRIPTION

The essd daemon provides a process context for the ARP agent to broadcast ARP requests to known hosts. Each host that supports ARP resolution over HIPPI directs its ARP request messages to HIPPI-SC logical address 0xfe0. The ARP agent receives the ARP request messages. Each message is forwarded to all the known hosts. This is a replacement mechanism for Ethernet broadcast. The daemon makes an ioctl call that sleeps in the kernel. The daemon cannot be killed. To stop the daemon execute "essd -k". Starting and stopping the daemon requires superuser privileges.

Starting and stopping the daemon requires superuser privileges.

OPTIONS

-h Set the given host name into the driver. This is used by the start-up scripts since hostname has not been set yet. After the system is booted this is not needed.

-k Kill the daemon. This is the only way to kill essd.

SEE ALSO

arp, essarp

5.3 esshippi

NAME

esshippi Essential HIPPI control and status utility.

SYNOPSIS

esshippi on [long|short] [fp|ph] [network|loopback] [switched|direct] [unit]

esshippi off [dump] [unit]

esshippi restart [dump] [unit]

esshippi status [unit]

esshippi accept [unit]

esshippi reject [unit]

esshippi version

esshippi cards

DESCRIPTION

The esshippi command is used to control the state of Essential HIPPI driver and hardware or to query the current status.

Use of this utility requires superuser privileges except for obtaining status.

OPTIONS

on — Activate the HIPPI device. This option must be used before the Essential HIPPI driver will be initialized for data traffic. If the system is to be used strictly as an IP network then the "short" parameter should be used. If the system will also share other HIPPI-FP and/or HIPPI-PH traffic of arbitrary length (i.e., greater than 64kB) or needing to use the connection control or packet control facilities the "long" parameter should be used. If the system is to be used connected to other systems then the "network" parameter should be used (this is the default) otherwise the "loopback" parameter should be used. If going through a switch use "switched" else use "direct". Each unit or card can be configured independently. The NIC must be "ifconfig down" to run this command.

off — Resets the Essential HIPPI hardware and places the system into a state where it cannot accept or transmit packets. All API calls will receive ENODEV errors until the esshippi on command has been issued.

status — Query the current system status. A flags word indicates whether the system is UP or DOWN. If the system is UP then other flags indicate if the system is accepting or rejecting connection requests and if the system will allow the transmission of long packets. Other statistics are:

SRC connections — Number of connections generated.

SRC packets — Number of packets sent.

SRC failures — Number of errors attempting to transmit packets. There is no breakdown of errors on transmission. In particular connection timeouts, connection rejects, and sequence errors are all counted under this one multipurpose counter.

DST packets — Number of destions packets received.

DST rcv on bad ulp Number of packets received destined for a non-active ULP.

DST hippi - le drop Number of packets dropped due to lack of resources in the IP stack.

DST data errors Number of packets received with data errors (either parity or LLRC).

DST sequence err Number of packets with HIPPI sequence errors.

DST sdic lost Number of times the interconnect signal dropped.

accept Set the system into a mode where it will accept incoming connection requests.

reject Set the system into a mode where it will reject incoming connection requests.

version Reports the driver and runcode version numbers.

cards Reports the number of NICs in the host sytem.

SEE ALSO essstat, Essential HIPPI API Reference

5.4 blast

NAME blast Essential HIPPI Character driver transmitter example

SYNOPSIS blast [-2PMrkC] [-D device_file] [-I ifield] [-l size] [-n writes-per-pass] [-m passes] [-u ULP]

DESCRIPTION Test program and sample code for testing and using the Essential HIPPI character device. This program includes most of the ioctl() settings for transmitting provided in the API (See the Essential HIPPI API Reference). It can also be used in conjunction with sink to get a rough system performance estimate.

Generally speaking, blast does writes-per-pass writes of size, passes times. So if you wanted to send 800 (eight hundred) 2MB packets 10 times you

would specify -l 0x200000 -n 800 -m 10. You can also combine the writes-per-pass into a single packet by using the -P flag (see below). So to send 10 1.6 GB packets you would specify -l 0x200000 -n 800 -m 10 -P.

OPTIONS

All of the arguments are optional and defaults are used specific values are not entered.

-2 If selected then the application will run double threaded (i.e., two processes will both write to the device at the same time).

-P Each pass will be put into a single packet.

-M This indicates that the buffers will be locked into memory with the mpin() system call.

-k This indicates that the buffers will not be allocated in user space. Instead, preallocated physically contiguous memory in the kernel will be mapped into user space and used. This increases throughput by nearly a power of two, but there are only two 2MB buffers available.

-r Random data will be used instead of the standard ASCII printable characters. This makes the checking option on sink(1) inoperative.

-D For platforms supporting multiple Essential HIPPI cards specifies the "base" device. Default is /dev/ess0.

-I Sets the Ifield for connection. Default is 0.

-l The size of the buffer passed to each write() call. Default is 2MB (0x200000 bytes).

-n The number of writes per pass. A pass is used to indicate when the packet is ended (for -P) or when the connection is dropped (for -C). The default is 500.

-m The number of passes. The default is 1.

-u The "upper layer protocol identifier" for the framing protocol header. Must match the one specified by sink(1). The default is 0x82.

SEE ALSO

sink

5.5 sink

NAME sink Essential HIPPI Character driver receiver example.

SYNOPSIS sink [-2] [-D device file] [-l size]
[-n number-of-reads] [-u ULP] [-c checking-level] -s

DESCRIPTION

Test program and sample code for testing and using the Essential HIPPI character device. This program includes most of the ioctl() settings for receiving provided in the API (See the Essential HIPPI API Reference). It can also be used in conjunction with blast to get a rough system performance estimate.

Sink accepts packets of whatever length in size chunks. It will run forever or until number-of-reads reads have been successfully completed, or until an error is encountered when in checking mode.

On Indigo^2 systems the buffers are mpin()ned to lock them into memory.

OPTIONS

All of the arguments are optional and (mostly) appropriate defaults are used if not set.

-2 If selected then the application will run double threaded (i.e., two processes will both read from the device at the same time).

-o If selected then the application will run in one-read mode. This means that the D1 data will not be split away from the D2 data as explained in the API document.

-k This indicates that the buffers will not be allocated in user space. Instead, preallocated physically contiguous memory in the kernel will be mapped into user space and used. This increases throughput by nearly a power of two, but there are only two 2MB buffers available.

-D For platforms supporting multiple Essential HIPPI cards specifies the "base" device. Default is /dev/ess0.

-l The size of the buffer passed to each read() call. Default is 2MB (0x200000 bytes).

-s If selected, the application will bind to the ULP using a shared bind. The default is to use an exclusive bind.

-n The number of reads done prior to exit. The default is "forever."

-u The "upper layer protocol identifier" for the framing protocol header. Must match the one specified by blast(1). The default is 0x82.

-c If the level is 1 then only the first data error will be flagged. If the level is 2 then all data errors will be flagged. The default is "no checking."

SEE ALSO blast

5.6 tsock

NAME tsock Socket-based network exerciser

SYNOPSIS tsock [-r] [-t] [-s] [-d[x]] [-lbuflen] [-nnbufs] [-Tseconds] [-mnby] [-?l] [-?s] [-?r] [-?R] [-?Nnbufs] [-pport] [-bsocketbufsize] [-h] [-ooffset] [-B] [-aabufs] [-cfreq] [-X] [-xtimes] [-ztimes] [-Pxpat] [-Ffilename] [-Sseconds] [hostname]

DESCRIPTION The tsock program uses the BSD socket interface to perform numerous network exercises. You can use it for several purposes, including timings, connectivity testing, fault diagnosis, and software checkout. The program is a derivative of the public-domain ttcp program developed by T.C. Slattery, USNA, and improved by Mike Muuss, BRL.

OPTIONS

-r Operates as a receiver.

-t Operates as a transmitter.

-s Operates in source/sink mode, in which the transmitter sends a standard pattern of internally generated data and the receiver discards all data. This avoids disk I/O (so it is useful for timings) and allows validation of incoming data. (See the -c option.) You can also specify alternate patterns can also be specified. (See the -P and -F options.)

-d Uses datagrams (ISO Connectionless Transport Service).

-lbuflen Transmit or receive buffer size, in bytes. The default buffer size is 1024 bytes. If you append a "K" or "k" to the value, it is multiplied by 1024. If you append an "M" or "m", the value is multiplied by 1048576. If you append a "G" or "g", the value is multiplied by 1073741824.

-nnbufs (For the transmitter only.) Transmits a total of nbufs messages. The default is 1024.

-Tseconds (For the transmitter only.) Transmits for the total time specified by seconds.

-mnby Mark the progress of the transfer by printing an "X" for every nby bytes sent or received. The default is 10240. If you append a "K" or "k" to the value, it is multiplied by 1024. If you append an "M" or "m", the value is multiplied by 1048576. If you append a "G" or "g", the value is multiplied by 1073741824. This option does not work well in conjunction with the -x or -z option.

-?l Uses a random-length buffer ranging from 1 byte to buflen. This option is applicable only in source/sink mode and, if specified for the receiver, for non-datagrams only; otherwise, it is ignored.

-?s Uses a random starting byte location within the buffer (that is, a random byte boundary). This option is applicable only in source/sink mode; it is ignored otherwise.

-?r Does not seed the random number generator. Specify this option to produce repeatable "random" runs.

-?R Fills the transmit buffer with random data.

-?Nnbufs Regenerates the random data in the transmit buffer every nbufs buffers. If you specify this option, you must also specify the -?R option.

-pport Port number for the connection or datagram path. The port argument must be a value between 1024 and 65535. There is a BSD convention that port numbers under 1024 are usable only by privileged programs. The default port number is 2000. Port numbers are 16 bits.

-bsocket-bufsize Sets the socket buffer size using the SO_SNDBUF or SO_RCVBUF socket option commands. This changes the internal buffer in the sockets portion of the kernel and may have a drastic negative effect the performance of the transfer.

-h Sets the SO_DEBUG socket option on the connection. This is useful for debugging connection problems.

-ooffset Changes the buffer offset the specified number bytes from the "natural" buffer boundary. On many systems, the buffer boundary is a page boundary.

-B Outputs full buflen-sized blocks only (for tar). This option is applicable for the receiver only; it is ignored for the transmitter.

-aabufs Uses asynchronous I/O with the specified number of simultaneous operations. You must specify a value for abufs; there is no default. This option is applicable only on systems that support asynchronous I/O; it is ignored otherwise.

-cfreq Check incoming message at the specified frequency for correctness. If you omit freq, every message is checked. This option is applicable only in source/sink mode and only for the receiver; it is ignored otherwise. Depending on the value of freq, this option can significantly slow down the data transfer.

-X Do not check for extra receive bytes,

-xtimes Runs the command the specified number of times in parallel by forking. Specifying this option for tsock -t multiplies the number of bytes transmitted by time. Specifying this option for tsock -r causes each receiver process to receive approximately 1/times of the bytes transmitted. (This is approximate because there is no coordination among the receivers. They are all in a big free-for-all fight for messages.) Specifying this option is not the same as using multiple tsock commands, because all participants share the same port numbers (that is, they share the same connection or datagram path). This option is most useful in source/sink mode because of the free-for-all nature of both the transmitting and receiving.

-ztimes Runs the command the specified number times in parallel by forking (same as -x), except that the forked processes alternate between the transmitter and receiver. For example, specifying -z4 results in two transmitters and two

receivers. This allows you to operate a connection or datagram path in true full-duplex mode (that is, with data flowing simultaneously in both directions). Several considerations must be kept in mind when using -z; see the "Usage" section below.

-Pxpat — Uses the alternate pattern specified by the hexadecimal string xpat. If you do not specify this option, the default pattern, which runs from 0x20 through 0x73 repetitively is used. If you specify -P but omit xpat, the alternate cyclical, which runs from 0x00 through 0xFF, is used. The xpat argument must contain an even number of hexadecimal digits (0-9, a-f, or A-F). It cannot start with 0x.

If you specify this argument, you must also specify the -s option. If you specify the -c option to enable data checking, you must specify the -s option on both the receiver and the transmitter. If you specify xpat, data checking works only if the transmitter and receiver have identical buffer sizes and you do not specify any -? arguments.

-Ffilename — Uses an alternate pattern from the specified file. If you specify this argument, you must also specify the -s option. If you specify the -c option to enable data checking, you must specify the -s option on both the receiver and the transmitter. Data checking works only if the transmitter and receiver have identical buffer sizes and you do not specify any -? arguments.

-Sseconds — Uses select with a timeout of seconds instead of going directly to accept(2). Specifying a timeout value of 0 polls, and a timeout value of -1 waits forever.

hostname — Name of the local (receiver) or remote (transmitter) host to use for the connection or datagram path. This argument is required for the transmitter and is optional for the receiver.

USAGE

First issue the receiver command (tsock -r) on the receiving system. Then issue the transmitter command (tsock -t) on the sending system. The receiver takes the passive role, and either listens for an incoming connection or issues a recvfrom() for an incoming datagram. The transmitter takes the active role, and either initiates a connection or starts sending datagrams. Once the transmitter has sent the specified number of messages (nbufs), it either closes the connection or sends a special "end of transmission" datagram. Both the transmitter and receiver then generate detailed timing information to stderr. (The use of stderr allows the receiver to redirect stdout to a file.)

Unlike sockets, message boundaries are significant for ISO Connection-Oriented Transport Service. This means that each message that is larger than the receive buffer satisfies more than one read(), but each message that is smaller than the receive buffer satisfies a separate read(). For example, assume a receive buffer of 1024 bytes. If the transmitter were to send a 2000 byte message (either because buflen were 2000 or by random choice), the receiver's first read() would return 1024 bytes and its second read() would return 976 bytes. If the transmitter were to send a 5 byte message, the 1024 byte read() would return only the five bytes.

The maximum size for a datagram is one TPDU (normally 32768 bytes), and datagrams are truncated rather than continued. In the first example above (receiver buflen of 1024, transmitter sends 2000 bytes), only the first 1024 bytes of the message (datagram) would be delivered; the remaining 976 bytes would be discarded.

An alternate pattern file (specified with the -F option) is an ASCII file containing a hexadecimal representation of the pattern to be repeated throughout each message; the file must be less than 1000 characters in length. Each valid hexadecimal character in the file (0-9, a-f, A-F) is converted into binary and included in the pattern, and all other characters are ignored in order to allow inclusion of tabs, spaces, carriage returns, and so forth. There must also be an even number of hexadecimal characters (the pattern is a string of bytes, not nibbles). For example, each of the following files would result in messages consisting of the 16 hexadecimal digits repeating throughout:

0123456789abcdef

01 23 45 67 89 AB CD EF

01/23 45/67 89/ab cd/ef

"01234567"

"89abcdef"

When using the -z option, you must specify values for nbufs on both commands. Otherwise, the transmitters and receivers will not know when to terminate; these may be specified either with the -n option or by using the default. Also, take care to balance the total number of transmissions and

receptions. Note that this does not mean simply that the two values of nbufs must be the same. For one thing, nbufs is essentially multiplied by the number of processes of each type. This is particularly bothersome when times is odd or is different on the two commands. Also, a single transmission may satisfy several receives (for example, one 3000 byte transmit satisfies four 800 byte receives). This is significant when buflen is different on the two commands. Because of this, the -z option used with random-length buffers (-?l) generally does not work correctly. It transfers data, but generally does not terminate. Finally, note that specifying the -z option makes it meaningful to specify the -c option along with the -t option.

EXAMPLES

The following pair of commands sends 1024 canned messages of 1024 bytes each (the default values) from the host mulberry to the host juniper:

On juniper: tsock -r -s juniper

On mulberry: tsock -t -s juniper

The following pair of commands sends 100000 random-sized messages (ranging from 1 byte to 50000 bytes) from the host mulberry to the host juniper. On juniper, the messages are received in a randomly located, random-sized buffer (ranging from 1 to 10000 bytes long); the messages are checked for correctness. These commands also mark every 100000 bytes received and use a maximum TPDU size of 8192 bytes. Note the use of backslash to bypass shell interpretation of the question mark.

On juniper: tsock -r -s -l10000 -\?l -\?s -c -m100000 -u13juniper

On mulberry: tsock -t -s -l50000 -\?l -n100000 juniper

The following pair of commands sends the file testdata from the host mulberry to the host juniper using 256 byte messages.

On juniper: tsock -r -l256 juniper > testdata

On mulberry: tsock -t -l256 juniper < testdata

The following pair of commands sends 1024 datagrams of 20 Kbytes (20480 bytes) each from the host mulberry to port 555 on the host juniper. On juniper, the datagrams are checked for correctness. In addition, an "X" is printed

on mulberry for every ten datagrams sent, and on juniper for every ten datagrams received.

On juniper: tsock -r -s -d -p5555 -l20k -m200k -c juniper

On mulberry: tsock -t -s -d -p5555 -l20K -m 200K juniper

The following pair of commands sends 1024 random-length datagrams of up to 20000 bytes each from the host mulberry to port 555 on the host juniper. Note that because the receive buffer is only 10000 bytes long, any datagrams longer than 10000 bytes are truncated. This should be about half the datagrams if the random number generator is reasonable.

On juniper: tsock -r -s -d -l10000 juniper

On mulberry: tsock -t -s -d -l20000 -\?l juniper

The following pair of commands sends 100 1024 byte messages from each of five parallel transmitters to four parallel receivers, using a single connection. A total of 512000 bytes are transmitted (1024 x -n100 x -x5). Note that each of the four receivers probably will not receive exactly one-fourth of them. However, the total bytes received will be 512000.

On juniper: tsock -r -s -x4 juniper

On mulberry: tsock -t -s -n100 -x5 juniper

The following pair of commands sends 200 1024 byte messages in each direction between juniper and mulberry (for 400 messages total), using a single connection. The data uses an alternate pattern and is checked on both ends.

On juniper: tsock -r -s -c -P0000ffff -n100 -z4 juniper

On mulberry: tsock -t -s -c -P0000ffff -n100 -z4 juniper

The following commands are identical to the previous pair, but use the alternate pattern specified in the file alt-pat4.

On juniper: tsock -r -s -c -Falt-pat4 -n100 -z4 juniper

On mulberry: tsock -t -s -c -Falt-pat4 -n100 -z4 juniper

DIAGNOSTICS

Error messages are self explanatory. Status and timing output is directed to stderr to allow the receiver to redirect stdout into a file. If invoked with no parameters, tsock prints a usage summary

BUGS

Until multiple-adapter support is implemented, whenever the configuration has more than one adapter defined, you must alwasy specify a hostname parameter for the receiver. (A hostname parameter is always required for the transmitter.)

In source/sink mode using datagrams, the sender produces datagrams as fast as possible, and in general will overrun the ability of the receiver to receive; some datagrams will usually be discarded. This situation is exacerbated by the use of the -c option. Occasionally the special "start of transmission" or "end of transmission" datagrams will be missed, resulting in a hung receiver. This situation is exacerbated by using the
-x option on a datagram receiver, which is really not recommended.

The option parser is relatively stupid, and requires that there be no spaces between an option letter and its argument (for example, -n1000 rather than -n 1000 and -?s rather than -? s).

5.7 essdmpd

NAME

essdmpd HIPPI dump Daemon

SYNOPSIS

essdmpd [-a] [-k] [-d <dump_dir>]

DESCRIPTION

This is a system daemon. The process forks and the parent dies. When the user requests that a dump file be generated, through "esshippi off dump" or "esshippi restart dump", and when the watchdog discovers that the NIC is not operating properly, the essdmpd daemon extracts the relevant information and produces a dump file as described below. The dump files are placed, by default, in /var/adm/hippi. The -d option permits the user to specify a different target directory. The target directory must already exist and root must be able to create and update files in the target directory. When the first dump file is generated a "Bounds" file is created. The "Bounds" file is

an ascii file that contains one line for each defined HIPPI card that has been dumped into the target dirctory. The fields in the line are decimal numbers saparated by a space. The line is terminated by a newline character. Each line contains the following fields:

"card_number" - number of the card that is being dumped.

"set_number" - the set number of the most recently created set.

"current_files" - number of sets in the directory for the card.

"max_sets" - maximum number of sets allowed for the card.

As dump requests are received, the daemon produces the requested file. When the maximum number of files limit is reached, the lowest number file for the card is deleted to make room for the new file. The default maximum number of files is 5. The "Bounds" file may be edited to change the number. All of the files, including the "Bounds" file, may be deleted at any time. Starting and stopping the daemon requires super-user privileges. There is one essdmpd daemon running for the host system. Daemon errors are logged via syslog in /var/adm/syslog.dated/<date>/daemon.log

OPTIONS

-a Produce an ASCII dump file instead of the default binary file. This option must be used with caution because the ASCII dump file can be quite large and it doesn't contain as much information.

-d <dump_directory>

Place dump files in this directory. The directory must already exist. Root must be able to create and modify files in the directory.

-k Kills the daemon.

SEE ALSO esshippi, essdisp

6

ADMINISTRATIVE UTILITIES

This chapter describes the administrative-level and debug-level diagnostic utilities included with the Essential HIPPI driver software.

Note: The diagnostics described in this appendix are not user-level diagnostics. Use them with extreme caution.

This appendix describes the following diagnostic utilities:

- **essdb** allows you to change the internal trace level.
- **essstat** displays internal statistics about the device driver.
- **esstb** copies the internal trace buffer to standard out.
- **esstune** is a performance tuning utility.
- **essgidle** is a GIO bus performance tuning utility.
- **essdnld** downloads firmware to the card.
- **essdisp** is a HIPPI driver objects display utility

6.1 essdb

NAME	essdb	Essential HIPPI driver debug trace display and control
SYNOPSIS	essdb [-t trace-level] [-d trace-level] [-v validation-level]	
DESCRIPTION	The essdb utility displays and controls the debug trace levels of the Essential HIPPI EISA driver.	

OPTIONS

no flags — Displays the current status of the type t tracing, the type d tracing, and the packet validation levels.

-t trace-level — Sets the type t tracing level. Level 0 turns off all type t tracing. Level 1 turns on the most general type t tracing. Level 2 turns on all type t tracing. The system normally runs with type t tracing set to 1. You must have superuser privileges to specify this option.

-d trace-level — Sets the type d tracing level. Level 0 turns off all type d tracing. Level 1 turns on the most general type d tracing. Level 2 turns on all type d tracing. The system normally runs with type d tracing set to 1. You must have superuser privileges to specify this option.

-v validation-level — Sets the level of validation for HIPPI packets. The default level is 0, which performs very minimal validation. Setting this to 1 turns on a great deal of validation. Level 1 may not be interoperable with all existing systems. If the status returns with a value of -1, the validation code is not turned on in the kernel. Therefore you cannot change the validation level. You must have superuser privileges to specify this option.

SEE ALSO — essarp, essstat, esstb

6.2 essstat

NAME — essstat — Essential HIPPI driver statistics display

SYNOPSIS — essstat [-c] [-g] [-d] [-i] [-a] [-s] [-x] [-f]

DESCRIPTION — The essstat utility displays internal statistics about the HIPPI device driver.

OPTIONS

-c — Display configuration information. Most of the information in this section is very device specific and won't mean much to the non-developer.

-g — Display data movement statistics and interesting adapter history statistics, such as number of times the adapter has been reset, number of short packets reassembled, etc.

-d Display data movement statistics for any open character devices.

-i Display statistics about IP data traffic.

-a Display all of the above statistics.

-s Display all NIC cached (non-zero) hardware statistics.

-x Display all current NIC (non-zero) hardware statistics.

-f Modify -s and -x arguments to display all statistics.

SEE ALSO essarp, essdb, esstb

6.3 esstb

NAME esstb Essential HIPPI driver debug trace display

SYNOPSIS esstb

esstb -n Display formatted version of firmware debug trace buffer.

esstb -n -D [unit] Used for multiple cards.

DESCRIPTION The esstb utility displays a formatted version of the debug trace buffer.

OPTIONS

-n Display a formatted version of the NIC firmware debug trace buffer.

-D Only used along with the n parameter. Used to specify which "unit" or card to display.

SEE ALSO essarp, essdb, essstat

6.4 esstune

NAME esstune Essential HIPPI driver debug performance tuning utility.

SYNOPSIS esstune [-c <retry count>] [-t <retry timer>] [-o <campon_timeout]

esstune [-s <stat_timer>] [-i <interrupt_timer>]

esstune [-x <tx_idle>] [-r <rx_idle>]

esstune [-w <dma_write_state>] [-d <dma_read_state>] [-h <pci_state_reg>]

esstune [-l] [-p] [-e] [-D <unit>]

DESCRIPTION

The esstune utility allows performance tuning for the HIPPI device driver.

OPTIONS

-c When the HIPPI-SC "Campon" bit is not set in the I-field, a rejected connection is retried this number of times before giving up. the counter can be set to do 0 or more retries.

-t When the HIPPI-SC "Campon" bit is not set in the I-field, a rejected connection is retried at this interval between retries.

-o When the HIPPI-SC "Campon" bit is set in the I-field, the adapter will wait for the connection to be accepted for the configured time.

-s The adapter keeps operating statistics and places a snapshot of the statistics in host memory at this configured interval. A value of 0 prevents the statistics from being automatically copied to host memory.

-i It is possible for the adapter to generate interrupts faster than they can be handled by the host system. This timer prevents back-to-back interrupts fro occurring too quickly by delaying the following interrupt for the configured time. Short times tend to flood the host with interrupts and lo9ng times tend to reduce the responsiveness of the device.

-x When the transmit connection has not passed any data in the configured time, the connection is aborted.

-r When the receive connection has not passed any data in the configured time, the connection is aborted

-w See the Roadrunner specification for bit settings.

-d See the Roadrunner specification for bit settings.

-h See the Roadrunner specification for bit settings.

-l -e Set the current esshippi flags into EEPROM.

-p Display the current values of the tuning parameters in the driver (-p) or the EEPROM (-p -e).

-e Make changes to the EEPROM. (NIC must be off)

-D The NIC unit must be used as the last parameter in multiple NIC configurations.

SEE ALSO esshippi

6.5 essgidle

NAME essgidle HIPPI driver GIO bus usage tuning utility.

SYNOPSIS essgidle value

DESCRIPTION Essgidle Controls access to the GIO bus. The HIPPI NIC and the graphics adapter share the GIO bus. This utility sets the relative performance of the graphics and HIPPI adapters by setting the number of cycles that the HIPPI NIC waits after releasing the bus before asking for the bus again. The value recommended by SGI (the default) is 128 (0x80). Lower values result in better HIPPI performance and worse graphics performance. On a system that is not using graphics, a value of 8 may be used to optimize HIPPI performance.

OPTIONS

-value The number of clock cycles to wait. Value is in decimal or hexadecimal (0x).

SEE ALSO esshippi, esstune

6.6 essdnld

NAME essdnld Essential HIPPI driver firmware download utility.

SYNOPSIS essdnld [-dc] [-D [unit]] [-l <file>] [-e<file>] [-r <file>]

DESCRIPTION This utility is used to manage the firmware download to the NIC. The firmware can be located in the EEPROM on the NIC or as a cached image in the memory space of the driver. The NIC must be stopped before using this utility. The utility requires super user permission.

OPTIONS If no parameters are specified, essdnld displays the help message. Command line options cannot be combined.

-d Delete the firmware in the driver cache.

-D Specify device for download, hip0 is the default.

-c Clear the runcode only from the EEPROM. Preserves manufacturing and tuning data.

-l Download a hex format runcode file to the driver memory cache.

-e Load a full firmware image from a hex formatted file into the EEPROM. Preserves existing serial number and ULA values.

-r Download a Runcode image only, from a hex formatted file, into the EEPROM. Preserves manufacturing and tuning data.

SEE ALSO esshippi, essstat

6.7 essdisp

NAME essdisp Essential HIPPI NIC objects display utility.

SYNOPSIS essdisp -D <dev> <general_options> <device_options>

essdisp -f <file> <general_options> <device_options>|<program_options>

<general_options> [-H] [-p] [-F] [-x]

<device_options> [-a] [-d] [-r] [-s] [-n] [-l] [-e] [-m] [-t] [-N]]

<program_options> [-A] [-L] [-S] [-T] [-P] [-V]

DESCRIPTION

Essdisp is a diagnostic utility that displays information is retreived from a NIC either directly by this utility or previously by the dump daemon (essd-mpd). It also displays the internal structure of RunCode program files.

OPTIONS

The -D <dev> form extracts information about a device <dev> and displays it. The list of hippi devices may be displayed by "netstat -i".

The -f <filename> form processes file <filename> and displays the the desired records. Files contain RunCode images or NIC dump images.

The [<general_options>] control the general operation of the utility.

The [<device_options>] are used either when directly accessing a NIC or displaying a NIC dump file.

The [<program_options>] are used when displaying a RunCode file.

If no options are specified, the command prints out a usage message.

General Options -

-H Displays record headers.

-p Displays the generally used "partial" information from the records.

-F Displays all "full" information from the records.

-x Displays the entire record in hex.

NIC DUMP OPTIONS

-a Displays all NIC dump records.

-d Displays driver records. These records contain data structures that are used by the driver to manage the NIC.

-r Displays all the ring records. The rings are the principle interface between the driver and the NIC. There are (up to) 256 receive rings, a send ring, an

event ring and a command ring. A list of descriptors, if any, is printed for each ring. The receive ring number corresponds to the 8 bit ULP number in the incoming packet.

-s Displays statistics for the driver and NIC.

-n Displays the NIC registers.

-l Displays the NIC SRAM contents.

-e Displays the NIC EEPROM contents. The NIC must be halted (esshippi off) to display the EEPROM.

-m Displays the manufacturing information area of the EEPROM. It shows the part number and revision for various components of the board, ULA address, board serial number, manufacturing data, etc. The NIC must be halted (esshippi off) to display the EEPROM.

-t Displays the driver trace buffer (same format as esstb).

-N Displays the NIC trace buffer (same format as esstb -n).

Program File Options -

-A Display all program file sections.

-L Displays all the "LINE" records, i.e. objects that have several line number records. The source-level debugger uses the source code line number and corresponding SRAM address information.

-S Displays all the symbol table records.

-T Displays all the text sections, i.e."TXT1", "TXT2" and "TEXT" for Phase 1 text, Phase 2 text and the Run Code text respectively.

-P Displays all the program counter record.

-V This option verifies the checksum if it follows a text segment.